

# Ancilla-based quantum simulation

Katherine L. Brown<sup>1</sup>, Suvabrata De<sup>1</sup>, Vivien M. Kendon<sup>1</sup>, William J. Munro<sup>2,3</sup>

<sup>1</sup>Department of Physics and Astronomy, University of Leeds, Leeds, LS2 9JT, UK

<sup>2</sup>NTT Basic Research Laboratories, NTT Corporation, 3-1 Morinosato-Wakamiya, Atsugi-shi, Kanagawa-ken 243-0198, Japan

<sup>3</sup>National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, Japan

**Abstract.** We consider simulating the BCS Hamiltonian, a model of low temperature superconductivity, on a quantum computer. In particular we consider conducting the simulation on the qubus quantum computer, which uses a continuous variable ancilla to generate interactions between qubits. We demonstrate an  $O(N^3)$  improvement over previous work conducted on an NMR computer [PRL **89** 057904 (2002) & PRL **97** 050504 (2006)] for the nearest neighbour and completely general cases. We then go on to show methods to minimise the number of operations needed per time step using the qubus in three cases; a completely general case, a case of exponentially decaying interactions and the case of fixed range interactions. We make these results controlled on an ancilla qubit so that we can apply the phase estimation algorithm, and hence show that when  $N \geq 5$ , our qubus simulation requires significantly less operations than a similar simulation conducted on an NMR computer.

## 1. Introduction

Quantum computing is a field of research which exploits the quantum behaviour of systems to get advantages over standard classical digital computing. The most famous example is Shor's algorithm, which gives an exponential improvement in factoring numbers compared to the best known classical algorithm [1]. However, even if we ignore the need for fault tolerance, such algorithms require thousands of qubits to obtain results inaccessible to classical computers and are therefore unlikely to be useful in the near future. A more accessible problem which is comparably hard to perform on a classical computer is simulating quantum systems. The idea of using quantum computers to simulate quantum systems was first proposed in 1982 by Feynman [2], and further developed by Lloyd [3] in 1996. Quantum simulation on a quantum computer would provide us with efficient algorithms to obtain: ground states of molecules [4]; details about bond formation [5]; and eigenvalues and eigenvectors [6] of many-body quantum Hamiltonians; for a recent review, see Brown et al. [7]. One of the largest systems of qubits which has been simulated in full generality on a classical computer consists of only 36 qubits [8]. By taking advantage of the symmetries that occur in many quantum

systems it is possible to simulate larger systems on a classical computer [9]. While 36 qubits is still a significant increase on what is currently available, it is an order of magnitude smaller than the break even point for many other quantum algorithms. Even with only tens of qubits it is possible to obtain interesting results that would be computationally intensive to obtain classically, especially when considering the general case scenarios. With more advanced and highly controlled quantum computers it will be possible to get savings on important simulations such as chemical reactions; so not only is quantum simulation a test bed for quantum computing, it will also significantly increase our understanding of many scientific problems.

One particular system of interest for an early simulation is the BCS Hamiltonian, a model of superconductivity formulated by Bardeen, Cooper and Schrieffer [10]. Since the BCS model of superconductivity is still poorly understood, a quantum simulation of the system could help rectify this situation. The same Hamiltonian is also used to describe pairing in nuclear physics, where it is called the pairing force Hamiltonian [11]. Pairing Hamiltonians are used to describe many processes in condensed matter physics and therefore a technique for simulating the BCS Hamiltonian should be adaptable to many other purposes. The BCS Hamiltonian,  $H_{BCS}$  is

$$H_{BCS} = \sum_{m=1}^N \frac{\epsilon_m}{2} (n_m^F + n_{-m}^F) + \sum_{m,l=1}^N V_{ml} C_m^\dagger C_{-m}^\dagger C_{-l} C_l \quad (1)$$

where  $C_m^\dagger$  and  $C_m$  are the Fermionic creation and annihilation operators,  $n_{\pm m}^F = C_{\pm m}^\dagger C_{\pm m}$  are the number operators,  $N$  is an effective state number that represents the number of modes to be simulated,  $V_{ml}$  is the interaction potential, and  $\epsilon_m$  is the on-site energy of a pair in mode  $m$ . Pairs of fermions have quantum numbers  $m$  and  $-m$  where pairs have equal energies but opposite momentum and spin,  $m = (\mathbf{p}, \uparrow)$  and  $-m = (-\mathbf{p}, \downarrow)$ .

The energy gap between the ground and first excited state of the BCS Hamiltonian is a non-perturbative function, this means that the Hamiltonian needs to be exactly diagonalised to obtain an accurate value for the energy gap. The non-perturbative nature of the Hamiltonian also means that to get an accurate value for the energy gap it is important to consider information from all interactions near the Fermi surface including long range interactions [12]. To get a solution for the energy gap, Bardeen et al. [10] took  $V_{ml} = -V$  for states near the Fermi surface, and 0 elsewhere. This approximation works for most metallic superconductors, but there are some cases where it is inaccurate. In these cases it would be useful to perform a simulation with generalised  $V_{ml}$ . While there have been attempts to do this classically, many classical approximation methods either don't take into account these long range interactions, or require them to obey very specific rules [13, 14, 15]. Those which apply to the general case, such as the Richardson solution [11], can only be solved numerically in a limited number of cases. Very recent results obtained by Ho et al. [16] obtain the energy gap of the BCS Hamiltonian without the need for the BCS approximation. However these results aren't as accurate for eigenvectors in the weak interaction regime. This means that to get an

accurate result, direct diagonalisation is often still necessary. However, it is resource intensive on a classical computer to diagonalise the exact Hamiltonians for more than a few tens of qubits. Therefore it is still difficult to characterise systems for cases where  $V_{ml}$  can't be held constant. Wu et al. [17] have shown that on a quantum computer it is possible to conduct a simulation of the BCS Hamiltonian efficiently, then use this simulation to extract information about the energy gap. While they concentrate on a nearest neighbour case, they also describe a case with general  $V_{ml}$ . Quantum simulation should thus be able to confirm whether the BCS Hamiltonian is correct for the metallic superconductors not covered by the assumption  $V_{ml} = -V$ .

To conduct a simulation of the BCS Hamiltonian on a quantum computer, we need to map equation (1) to the qubit Hamiltonian. This mapping is worked out by Wu et al. [17] and results in the BCS Hamiltonian taking the form:

$$H_{\text{pair}} = \sum_{m=1}^N \frac{\varepsilon_m}{2} \sigma_{zm} + \sum_{m<l}^N \frac{V_{ml}}{2} (\sigma_{xm} \sigma_{xl} + r \sigma_{ym} \sigma_{yl}) \quad (2)$$

where  $\sigma_{xk}/\sigma_{yk}$  are the Pauli  $X/Y$  on the  $k^{\text{th}}$  qubit, and  $\varepsilon_m = \epsilon_m + V_{mm}$  and  $r$  is a parameter determined by the mapping.

As we are interested in determining the energy gap between the ground and first excited state, we start by preparing our system in a superposition of these two states. This initial state is prepared using the technique proposed by Wu et al. for preparing an initial state on an NMR computer [17]. We first prepare our system of  $N$  qubits in a state of all zeros, this is a basic ground state. We then apply spin flips so we have a basis state containing  $n$  ones, where  $n$  is the number of excitations in the BCS Hamiltonian. This is different from  $N$  which is the number of modes in the BCS Hamiltonian and the number of qubits in our quantum computer. Using a quasi-adiabatic method of applying our Hamiltonian we prepare a superposition of the ground and first excited state, we discuss this step in more detail in section 4. This generates an initial state which will allow us to extract data about the low-lying energy spectrum of our Hamiltonian, and thus determine the energy gap.

To perform the time evolution and extract the data we use the phase estimation algorithm [18], this involves performing the BCS Hamiltonian controlled on a set of ancilla qubits for a set of time intervals given by  $2^{k-1}t$  for the  $k^{\text{th}}$  ancilla qubit. This encodes the data about the time evolution of the BCS Hamiltonian into the ancilla qubits. A quantum Fourier transform is then used to extract data from the system. In section 5 we looked at the number of operations required to implement our unitaries in a controlled fashion, and in section 6 we consider the number of operations needed to perform our quantum Fourier transform.

We will consider conducting our simulation on an ancilla-driven quantum computer, where an auxiliary system is used to generate interactions in the main processing unit. The main advantage of such a system is that it allows us to generate entanglement between distant qubits without the need for swap gates. In terms of the BCS Hamiltonian this facilitates the simulation of long range interactions, something difficult

to do classically. While our results are applicable to ancilla driven qubit computers in general we will concentrate in particular on the qubus quantum computer. The qubus is a continuous variable ancilla that is used to generate interactions between the qubits [19, 20]. The continuous variable bus is advantageous because detecting and controlling single photons is experimentally hard, while the continuous variable is easier to control. The qubus system generates gates between qubits deterministically without the need for measurement, but does require us to be able to generate an interaction between the field and a matter qubit [20].

In this paper we describe a method for simulating the time evolution of the BCS Hamiltonian then extracting the energy gap, using a similar technique to the one proposed by Wu et al. [17]. We show that using a qubus quantum computer gives significant advantages over an NMR quantum computer, and go on to show how we can obtain significant improvements over a naïve qubus implementation. This allows us to determine information about the ground and the first excited states of the Hamiltonian more efficiently than Wu et al. [17]. The simulation is limited in accuracy only by the number of terms used in the Trotter approximation to combine the non-commuting parts of the Hamiltonian, and the number of ancilla qubits. In both cases a linear increase in precision costs a linear increase in resources [21]. Other than this, no approximations are used and we can obtain results in the completely general case where the coupling between pairs can take any value for any pair. We can therefore access regimes unobtainable using classical approximation methods.

The paper is arranged as follows. In section 2 we introduce the qubus quantum computer and discuss how we go about reducing the number of interactions with the bus needed to perform certain operations. Section 3 contains three methods for producing the necessary unitaries for the BCS Hamiltonian on the qubus; subsection 3.1 concentrates on the general case; subsection 3.2 looks at a case where it is possible to get an order of  $N$  saving in return for a reduction of generality; and subsection 3.3 looks at fixed range interactions. Section 7 puts the results from sections 4, 5, and 6 together to work out how many operations are needed as part of the longest single run of the qubus computer. We summarise our results in section 8.

## 2. The Qubus Quantum Computer

A qubus quantum computer is a hybrid system consisting of a processing unit made of qubits and a continuous variable field ‘bus’ which generates interactions and transfers information between the qubits. Using the bus to generate interactions between distant qubits removes the need to either change calibration settings every time a new qubit is added or use swap operations to move qubits next to each other. There are several proposals for physical architectures which could potentially be used to build a qubus system. These include optical quantum systems [19] and super-conducting systems [22].

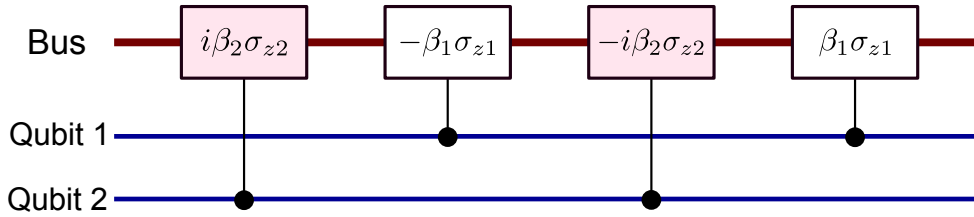
Interactions in the qubus architecture take the form of displacement operators

applied to the continuous variable field. These can be written in the form [20]

$$D(\beta\sigma_{zk}) = \exp(\beta\sigma_{zk}a^\dagger - \beta^*\sigma_{zk}a) \quad (3)$$

where  $a^\dagger$ ,  $a$  are the creation and annihilation operators,  $\sigma_{zk}$  is the Pauli Z operator acting on the  $k^{\text{th}}$  qubit,  $\beta = \chi te^{i(\phi - \frac{\pi}{2})}$ , and  $\chi$  is the strength of the nonlinearity being used. In the qubus system we use  $\phi = 0$ , corresponding to the position quadrature or  $\phi = \frac{\pi}{2}$ , corresponding to the momentum quadrature.

The displacement operators entangle the bus with the qubits. A qubit is entangled to either the position or momentum quadrature of the bus, and since these displacements are in orthogonal directions it is possible to create a maximally entangling gate. Using two qubits and one bus it is possible to generate a deterministic C-Phase gate by performing just four displacement operations on the bus. This scheme is illustrated in figure 1 and is given by the operator  $D(i\beta_2\sigma_{z2})D(\beta_1\sigma_{z1})D(-i\beta_2\sigma_{z2})D(-\beta_1\sigma_{z1})$ , see [20].



**Figure 1.** A circuit diagram for implementing the C-Phase gate on the qubus. The boxes show displacements performed on the continuous variable field controlled by the qubits as in equation (3). Shaded boxes represent an operation acting on the position quadrature of the bus, while unshaded boxes represent operations on the momentum quadrature.

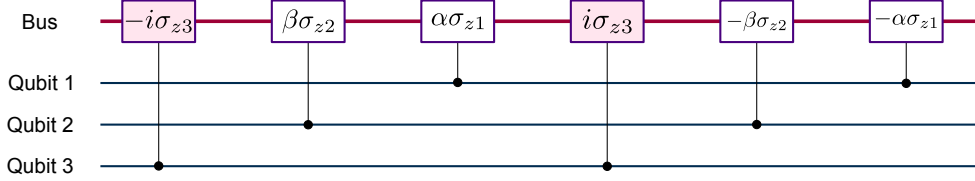
These operations are combined using the formula

$$D(a)D(b) = \exp\left(\frac{ab^* - a^*b}{2}\right) D(a + b) \quad (4)$$

where  $a$  and  $b$  are Pauli operators on the qubits and must commute (e.g. because they act on different qubits). This sequence thus results in the interaction  $\exp(2i\beta_1\beta_2\sigma_{z1}\sigma_{z2})$ , which when  $2\beta_1\beta_2 = \pi/4$  is equivalent under local unitaries to a C-Phase gate.

We will define the ‘naïve’ method of implementing an operation on a qubus to be implementing a Hamiltonian by performing each C-Phase gate individually, disentangling the bus from the system between each set of gates. This technique requires four operations for each C-Phase gate within the desired interaction. If we take an interaction of the form  $\exp(i\alpha\sigma_{z1}\sigma_{z3} + i\beta\sigma_{z2}\sigma_{z3})$ , then using the naïve method would require a total of eight operations to implement it. However, the term  $\sigma_{z3}$  appears twice, therefore it is possible to reduce the total number of bus operations by rearranging the gate sequence, so the 3<sup>rd</sup> qubit only needs to be interact with the bus twice. The gate sequence required to do this is shown in figure 2, it uses only six operations.

Provided there are ‘overlapping’ terms such as the  $\sigma_{z3}$  in the above example, it is possible to get a reduction in the number of operations required over the naïve method. The extent of this reduction depends upon the number of terms that overlap, how often these overlapping terms appear, and the level of generality required. We will now apply these techniques to reduce the number of operations needed to simulate the BCS Hamiltonian.



**Figure 2.** A reduced gate sequence for  $\exp(i\alpha\sigma_{z1}\sigma_{z3} + i\beta\sigma_{z2}\sigma_{z3})$  using 6 operations instead of 8. The boxes show displacements performed on the continuous variable field controlled by the qubits. Shaded boxes represent an operation acting on the position quadrature of the bus, while unshaded boxes represent operations on the momentum quadrature.

### 3. Implementing the Hamiltonian

We will now consider how to perform the BCS Hamiltonian on our qubus quantum computer, this stage is essential for both the initialisation process, where it is used for a semi-adiabatic evolution, and the time evolution stage. The time evolution of the BCS Hamiltonian in terms of Pauli operators is shown in equation (2), which consists of three non-commuting Hamiltonians so that  $H_{\text{pair}} = H_0 + H_{xx} + H_{yy}$

Taking the corresponding unitaries to these Hamiltonians we get

$$U_0 = \exp\left(i \sum_m^N \frac{\varepsilon}{2} \sigma_{zm}\right) \quad (5)$$

$$U_{xx} = \exp\left(i \sum_{m<l}^N \frac{V_{ml}}{2} (\sigma_{xm}\sigma_{xl})\right) \quad (6)$$

$$U_{yy} = \exp\left(i \sum_{m<l}^N \frac{V_{ml}}{2} r (\sigma_{ym}\sigma_{yl})\right). \quad (7)$$

Since these are non commuting unitaries,  $\exp(H_{\text{pair}}) \neq \exp(H_0)\exp(H_{xx})\exp(H_{yy})$ . To get around this we use the Trotter approximation [23, 24] splitting the time  $t$  into small segments  $\tau$ . In the first order case this is given by  $\exp(H_{\text{pair}}) \approx \exp(H_0\tau)\exp(H_{xx}\tau)\exp(H_{yy}\tau) + O(\tau^2)$ , and in the second order case it is given by  $\exp(H_{\text{pair}}) \approx \exp(H_0\tau/2)\exp(H_{xx}\tau/2)\exp(H_{yy}\tau)\exp(H_{xx}\tau/2)\exp(H_0\tau/2) + O(\tau^3)$ . Longer time intervals are built up by performing the short time evolution multiple times.

In equation (5),  $U_0$  is a sum of local operations and can be implemented using  $N$  local unitaries. As we are considering a situation where we have individual addressability

of the qubits, this is straight forward and we won't discuss it further. In equations (6) and (7),  $U_{xx}$  and  $U_{yy}$  are equivalent under local unitaries to

$$U_{zz} = \exp \left( i \sum_{m < l}^N \frac{V_{ml}}{2} (\sigma_{zm} \sigma_{zl}) \right). \quad (8)$$

We now consider how to implement equation (8) in three different cases.

### 3.1. The fully generalised case

Equation (8) is a sum of  $(N^2 - N)/2$  terms, each requiring one gate of the form  $\exp(i\beta_m\beta_l\sigma_{zm}\sigma_{zl})$ . For the simplest possible implementation on a qubus computer, we perform each of these gates individually requiring 4 bus operations per gate. This means  $U_{zz}$  requires a total of  $2N^2 - 2N$  bus operations to perform.

This technique uses the simplest possible procedure for generating the necessary interactions, and it should be possible to get reductions by reusing our bus. However, if we wish to be able to set each of our constants,  $V_{ml}$ , separately then we have significant constraints on how we can do this. A simple lower bound on the number of bus operations needed can be found by considering the number of individual constants. For a system being modelled on  $N$  qubits, we require  $(N^2 - N)/2$  constants, which have no dependence upon each other. Since our operations on the bus occur in pairs, and each pair can only produce one constant we can see that it would be impossible to generate this using less than  $N^2 - N$  bus operations.

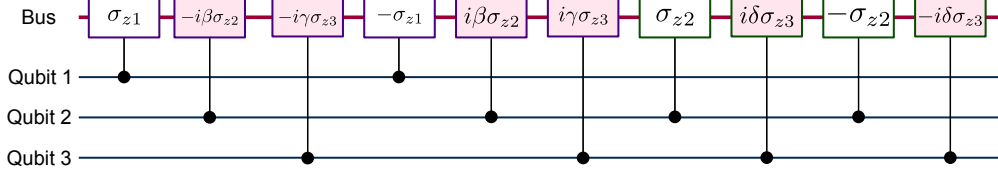
We now consider how we would go about generating the necessary bus operations. We want qubit 1 to interact with all the other  $N - 1$  qubits, with completely independent constants. The simplest way to do this would be to connect qubit 1 to one quadrature of the bus, then connect the other  $N - 1$  qubits to the other quadrature of the bus. Now, when we consider qubit 2, we can see that we have already generated the interaction between this and qubit 1, therefore it only needs to interact with the remaining  $N - 2$  qubits. However we want the constants generated from these interactions to be completely independent from the ones generated by interacting qubit 1 with the others. If we hold too many qubits on the bus, this independence isn't possible, therefore after each step we need to disconnect all of the qubits from both quadratures of the bus.

We therefore consider an  $N - 1$  step process, where for the  $a^{\text{th}}$  step we connect qubit  $a$  to one quadrature of the bus, and the other  $a - 1$  qubits to the other quadrature before disconnecting qubit  $a$  then all the others. The total number of bus operations needed for this is given by

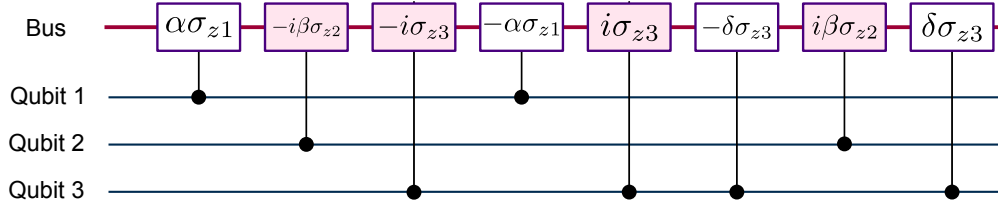
$$\sum_{a=2}^N 2a = 2 \left( \sum_{a=1}^N a \right) - 2 = N^2 + N - 2. \quad (9)$$

A circuit for generating interactions between 3 qubits using this technique is shown in figure 3.

One way of reducing the number of operations further is to leave some qubits on the bus at the end of each step. We now consider the case where we first attach qubit 1



**Figure 3.** A circuit for simulating  $U_{zz}$  for 3 qubits with generalised constants  $V_{ml}$  in 10 operations. The boxes show displacements performed on the continuous variable field controlled by the qubits. Shaded boxes represent an operation acting on the position quadrature of the bus, while unshaded boxes represent operations on the momentum quadrature.



**Figure 4.** A circuit for simulating  $U_{zz}$  for 3 qubits with generalised constants  $V_{ml}$  in 8 operations. The boxes show displacements performed on the continuous variable field controlled by the qubits. Shaded boxes represent an operation acting on the position quadrature of the bus, while unshaded boxes represent operations on the momentum quadrature.

to the first quadrature of the bus and the other  $N - 1$  qubits to the second quadrature, as before. However when we come to disentangle the qubits, we leave qubit 2 entangled to the second quadrature of the bus. Step two then involves attaching qubits 3 through to  $N$  to the first quadrature of the bus thus generating entanglement between all these qubits and qubit 2. In the disentangling step all the qubits are removed from the bus except qubit 3, which is used for step three. In this scenario qubit 1 and 2 interact with the bus twice (one connect and one disconnect) each, qubit 3 interacts with the bus four times (two connects and two disconnects) etc. This occurs all the way up to qubit  $N$  which interacts with the bus  $2(N - 1)$  times. This leads to a total number of operations given by

$$2 \left( \sum_{a=1}^{N-1} a \right) + 2 = N^2 - N + 2. \quad (10)$$

This is  $2N - 4$  operations less than our previous result and very close to our lower bound. A circuit for performing this technique using 3 qubits is shown in figure 4. It is less obvious that in this case the constants can be completely general as one constant from each step will carry through to the next step. However, since all the constants in the next step can be set arbitrarily it is possible to achieve this. If there is no interaction between two qubits, e.g. between qubits 2 and 3, then the higher numbered qubit is



skipped in that step and added in during a later step. This will never require extra operations and will in some cases provide a saving.

### 3.2. A limited case

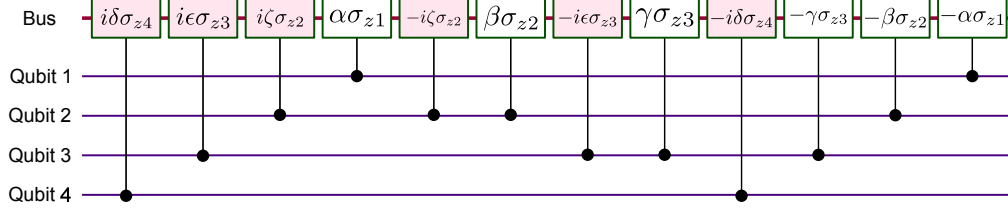
While the completely general case is important, many physical systems do not require this level of generality. If we are prepared to restrict our constants so that they become dependent upon each other then it is possible to get savings of  $O(N)$  over our previous techniques. This provides a saving for a range of different scenarios including ones where the strength of the interaction drops off exponentially with distance. We will begin by considering the simplest possible case of wanting to create the  $U_{xx}$  and  $U_{yy}$  that are local unitaries away from  $U_{zz}$  in equation (8) when  $V$  has no dependence upon either  $m$  or  $l$ . The minimal number of bus operations needed to generate this can be worked out by noting that for the sum given, both  $m$  and  $l$  can take  $N - 1$  possible different values. This means that  $2(N - 1)$  bus operations are needed to cover all the possible values of  $m$  and similarly for  $l$ . This results in a total number of bus operations given by  $4(N - 1)$ .

A method of performing the operations that meets this bound involves attaching qubits 2 through to  $N$  to the first quadrature of the bus. Qubit 1 is then attached to the second quadrature, generating entanglement between it and all the other qubits. Thus far, this matches the method in section 3.1. However now, instead of removing all the qubits from the bus, we disconnect qubit 2 from the first quadrature of the bus and then reattach it to the second quadrature. This generates entanglement between it and qubits 3 through to  $N$ . We continue this process, removing qubits 2 through to  $N - 1$  from the first quadrature, then attaching them to the second quadrature. Finally when we come to remove qubit  $N$  from the bus, we don't reattach it to the second quadrature. This leaves the first quadrature of the bus completely empty and  $N - 1$  qubits entangled to the second quadrature of the bus. The final step involves removing all remaining qubits from the second quadrature of the bus. This means every qubit interacts with the bus 4 times (2 connections and 2 disconnections), apart from the first and last qubit, which only interact with the bus twice. The total number of bus operations needed is given by  $4N - 4$ , therefore this technique meets the lower bound exactly. A circuit showing the sequence of bus operations for this technique in the four qubit case is shown in figure 5.

This technique can give constants more general than a fixed  $V$ . The constants attached to each qubit give a matrix of values for  $V$  that are dependent upon  $m$  and  $l$  and are linked as shown below

$$\begin{array}{cccc}
 & 2 & 3 & 4 \\
 1 & \alpha\zeta & \alpha\epsilon & \alpha\delta \\
 2 & & \beta\epsilon & \beta\delta \\
 3 & & & \gamma\delta.
 \end{array}$$

The entries of the matrix above represent the strength of the interaction between the



**Figure 5.** A circuit for simulating  $U_{zz}$  with a strong dependence between the constants. The boxes show displacements performed on the continuous variable field controlled by the qubits. Shaded boxes represent an operation acting on the position quadrature of the bus, while unshaded boxes represent operations on the momentum quadrature.

numbered qubits given the constants in figure 5, so for example the second column of the first row represents the strength of the interaction between qubit 1 and qubit 3. By looking at the matrix we can see that it wouldn't be possible to set values for  $\alpha, \beta, \gamma, \delta, \epsilon$  and  $\zeta$  such that all six entries of the matrix could be set completely arbitrarily. We can see this easily because  $\alpha\epsilon/\alpha\delta$  has to equal  $\beta\epsilon/\beta\delta$  if  $\alpha, \beta, \delta$  and  $\epsilon$  are complex constants; however if we set all 6 entries of our matrix randomly then this isn't always the case.

This dependence allows numerous cases, including an exponential fall off with distance or other more complicated dependencies. An example of exponential decay would be setting  $\alpha = 1, \beta = 2, \gamma = 3, \delta = 1/4, \epsilon = 1/2$  and  $\zeta = 1$ . The possible constants provide an interesting set of cases relevant to realistic situations.

### 3.3. Fixed range interactions

One case not covered by our limited case, in which it is obviously possible to get improvements, is nearest neighbour interactions. This would consist of a chain of qubits and is explored by Louis et al. [25] in the context of building cluster states. This involves a  $U_{zz}$  of the form

$$U_{zz} = \exp \left( \sum_{m=1}^{N-1} \frac{V_{m,m+1}}{2} (\sigma_{zm} \sigma_{z(m+1)}) \right). \quad (11)$$

Louis et al. find that for a chain of  $N$  qubits, it is possible to generate nearest neighbour interactions using just  $2N$  bus operations. This can be expanded to include a  $U_{zz}$  with both nearest neighbour and next-to-nearest neighbour interactions, a scenario which would require  $4N - 4$  bus operations.

It is possible to derive a general result for fixed range interactions by introducing a new variable  $p$ , where  $p$  represents the range of the interaction. When  $p = 1$  we consider only nearest neighbour interactions, when  $p = 2$  we consider nearest neighbour and next-to-nearest neighbour, etc.. We split our terms into a cycle of several steps, where we leave one qubit active with the bus at the end of each step, this allows us to keep the interaction strengths between various qubits completely generalised. For the

first step,  $p + 1$  qubits need to interact with the bus twice, this represents qubit 1 and the  $p$  qubits connected to it. The next set of steps need  $p$  qubits to interact with the bus twice, since we assume the 1<sup>st</sup> qubit in each step is already connected to the bus we only need to attach its  $p$  nearest neighbours. We require  $2p$  qubits operation on the bus for every single step except for the very first one and the  $p$  final steps. We therefore need to perform  $2p$  operations a total of  $N - p - 1$  times. Finally for the last set of steps there are no longer  $p$  qubits left in the chain connected to our active qubit, this means the number of bus operations required reduces from  $2(p - 1)$  to 0 in intervals of 2. Therefore, for an interaction length  $p$ , the number of operations on the bus needed to generate  $U_{zz}$  is given by

$$2(p + 1) + 2p(N - p - 1) + 2 \sum_{a=1}^p (p - a) = 2pN - p^2 - p + 2. \quad (12)$$

We can check that this equation agrees with our previous results. If we take the nearest neighbour case, then  $p = 1$  and we find that the total number of operations needed to generate  $U_{zz}$  is  $4N$ . If we take the case where all the qubits interact with each other,  $p = N - 1$  therefore  $N^2 - N + 2$  bus operations are needed to generate  $U_{zz}$ . This demonstrates a technique that allows the generation of interactions with a cut off at an arbitrary distance. It should be possible to obtain further savings in these cases using a technique similar to that described in section 3.2.

#### 4. Initialisation

To initialise our system we need to generate a superposition of the ground and first excited state for our BCS Hamiltonian. To do this, we can use the same method as Wu et al. [17] for the initialisation procedure within an NMR system. We begin by preparing our system so that we have  $N$  qubits all in state  $|0\rangle$ , where  $N$  is the number of modes in the BCS Hamiltonian, this is a basic ground state. By applying spin flips it is possible to transform this state so that  $n$  qubits are in state  $|1\rangle$ , where  $n$  is the number of excitations within our BCS Hamiltonian. This generates a computational basis state and the process is trivial to perform on a qubus quantum computer.

To get our qubus quantum computer to be in a combination of the ground and first excited state of the BCS Hamiltonian we need to implement an adiabatic form of our time evolution given by

$$U_{ad}(k\tau) \approx e^{-H(k\tau)\tau} \dots e^{-iH(2\tau)\tau} e^{-H(\tau)\tau} + O(\tau^2) \quad (13)$$

where  $\exp[-iH(j\tau)\tau] \approx \exp(iH_0\tau) \exp[-ic(j\tau)H_1\tau]$ ,  $j = 1, \dots, S$ ,  $S\tau = T$ , and  $c$  is a function which varies slowly with time,  $t$ , that goes from 0 at  $t = 0$  to 1 at  $t = T$ . In this case  $H_1 = H_{xx} + H_{yy}$ . We are gradually increasing the strength of the interactions within the Hamiltonian so that we go from a basis state of  $H_0$  to a state that is described by the BCS Hamiltonian. If we conducted this evolution in accordance with the adiabatic condition ( $S \gg \pi/(\tau\Delta)$  where  $2\Delta$  is the energy gap between the ground and first excited state), then we would end up in the ground state of the BCS Hamiltonian. However,

instead we relax the adiabatic condition and as a result end up with a component of the first excited state mixed in with the ground state. In the case of the BCS Hamiltonian the short time approximation is valid when  $\tau \ll 1/d$  where  $d$  is the level spacing, and  $\tau$  the length of a single time interval. This gives  $S \gg \pi d/\Delta$ . Brown et al. [21] add in another factor which represents the precision of the desired energy gap, and is given by  $\delta$ . This results in the condition  $S = \pi d/\delta\Delta$ . Taking  $d/\Delta = 0.1$  from Wu et al. [17] then the total number of time steps required is given by

$$S = \pi/\delta. \quad (14)$$

The time evolution here can be implemented using the method described in section 3. As we are considering a short evolution we can consider using only the first order Trotter approximation, similar to Wu et al. [17]. This means  $H_0$  can be implemented using single local unitaries on each qubit and  $H_1$  will require twice the number of operations needed to perform  $H_{zz}$  plus  $4N$  local unitaries to transform our  $U_{zz}$  terms to  $U_{xx}$  and  $U_{yy}$ . We can therefore see that the number of operations per time step,  $I(N)$  will be:

$$I_G(N) = 2N^2 + 3N + 4 \quad (15)$$

in the general case,  $I(N) = 13N - 8$  in the limited case and

$$I_L(N) = 4pN + 5N - 2p^2 - 2p + 4 \quad (16)$$

in the case of fixed range interactions. To work out the total number of operations needed for the initialisation procedure we multiply  $I(N)$  by  $S$ .

## 5. The Phase Estimation algorithm

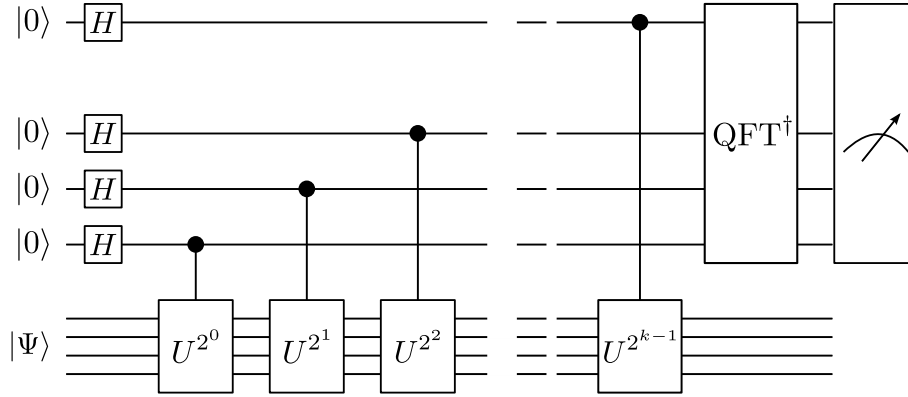
Once we have created the superposition of the ground and excited state we want to use the phase estimation algorithm for our data extraction procedure. The phase estimation algorithm has two stages, the first stages implements the unitary representing the BCS Hamiltonian and therefore encodes the data about our evolution into a series of ancilla qubit, the second uses a quantum Fourier transform to extract the data.

A circuit for performing the phase estimation algorithm is shown in figure 6. To illustrate the workings of the phase estimation algorithm we will consider our first register to be in a single eigenstate,  $|u\rangle$ , of  $U$ , therefore implementing  $U$  will leave  $|u\rangle$ . Implementing  $CU^{2^j}$ , i.e. controlled on an ancilla qubit results in

$$CU^{2^j} \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) |u\rangle = \frac{1}{\sqrt{2}} (|0\rangle + e^{i2^j\Phi}|1\rangle) |u\rangle. \quad (17)$$

where  $e^{i\Phi}$  given by  $U|u\rangle = e^{i\Phi}|u\rangle$  is the eigenvalue of  $|u\rangle$ . We now consider applying these gates from  $j = 0$  to  $j = k - 1$  using  $k$  ancilla qubits, with the net result

$$\frac{1}{\sqrt{2^k}} (|0\rangle + e^{i2^{k-1}\Phi}|1\rangle) \dots (|0\rangle + e^{i2\Phi}|1\rangle) (|0\rangle + e^{i\Phi}|1\rangle) |u\rangle = \frac{1}{\sqrt{2^k}} \sum_{y=0}^{2^k-1} e^{i\Phi y} |y\rangle |u\rangle \quad (18)$$



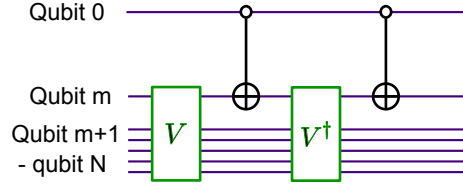
**Figure 6.** A circuit to perform the phase estimation algorithm, the sequence of controlled unitaries encode phase information from the input state  $|\Psi\rangle$  in the ancilla which is then extracted using the QFT.

Tracing out the register containing the eigenstate leaves us with our  $k$  ancilla qubits in the state  $\sum_{y=0}^{2^k-1} e^{i\Phi y} |y\rangle$ . Applying the inverse quantum Fourier transform encodes the phase into the register of ancilla qubits [26].

Assuming our register starts in an exact eignestate the final result of the phase estimation procedure is  $|\tilde{\Phi}\rangle|u\rangle$ , where  $|\tilde{\Phi}\rangle$  is a  $k$  qubit approximation of  $|\Phi\rangle$ . If the first register starts in an approximation of the eigenstate, then the phase estimation algorithm results in the second register being transformed into  $|\Phi\rangle$  with a probability that increases with the accuracy of the eigenstate and with  $k$  [26]. Similarly if the register starts in a superposition of eigenstates, then the ancilla register ends up being in one of the resultant eigenvalues with a probability that depends on how much the corresponding eigenstate contributes to the initial input state. In our case, since we start in a superposition of the ground and the first excited state we can extract the eigenvalue for the ground state, and the eigenvalue for the first excited state by using several runs of the computation. This will allow us to work out the energy gap.

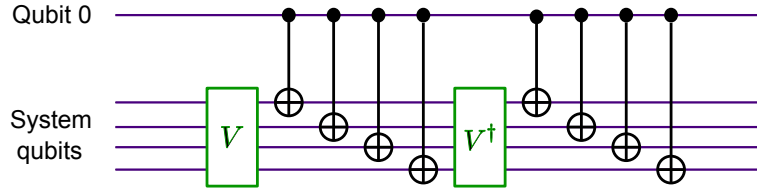
To implement our unitary as part of the phase estimation algorithm involves performing our operations in a controlled fashion, dependent upon the state of an ancilla qubit. In section 3 we discussed various techniques for performing our unitary,  $U_{zz}$ . We note that it is possible to make our  $U_{zz}$  controlled using a very simple control sequence such as the one shown in figure 7, provided each pair of operations acts on one common qubit. Since we need to perform CNOT operations after our sequence we can see the technique outlined in section 3.1 where we don't leave qubits connected to the bus, is ideal for use in this scenario.

Previous work on the qubus has outlined how to use the architecture to generate CNOT gates [20], each gate requires 4 bus operations and 2 local unitaries. We also note that it is possible to transform our operations  $U_{zz}$  to the form  $U_{xx}$  or  $U_{yy}$  using two local operations per qubit. Therefore we can work out the total number of operations needed to generate our two qubit interactions in a controlled fashion. In section 3.1 we split generating our  $U_{zz}$  into cycles where we interacted each qubit with all qubits numbered



**Figure 7.** A circuit to make a unitary  $U$  controlled on an ancilla qubit. In this case  $U = V^2$ , and  $U$  consists of pairs of Pauli  $Z$  operations where each pair of operations acts on qubit  $m$ .

higher than it. We now want to consider how many operations would be required to implement these cycles in a controlled fashion. A cycle which interacts qubit  $m$  with all the qubits numbered higher than  $m$  required  $2(N - m + 1)$  operations to generate. To implement this in a controlled fashion we use a control procedure such as the one shown in figure 7, therefore the total number of operations required per cycle is  $4(N - m + 4)$  including the gates required for our CNOT operations. To implement  $U_{zz}$  we will require  $2(N^2 + 7N - 8)$  operations. To transform  $U_{zz}$  to  $U_{xx}$  or  $U_{yy}$  requires an additional 2 operations per qubit bringing the total to  $2(N^2 + 8N - 8)$ .



**Figure 8.** A sequence to make  $N$  local unitaries controlled on an ancilla qubit, where  $V^2 = U_{L1} \otimes \dots \otimes U_{LN}$ .

The most efficient technique for making our local unitaries controlled involves performing our CNOT operations in a sequence, as in the circuit in figure 8, as a result a sequence which requires a single local unitary on each qubit will need  $8N + 4$  operations.

When combining our unitaries  $U_{xx}$ ,  $U_{yy}$  and  $U_0$ , we use the Trotter approximation. Since we need a more accurate  $U$  than for initialisation we will use the second order Trotter approximation [23, 24] which is given by

$$U_{BCS} \approx [U_0(\tau/2)U_{xx}(\tau/2)U_{yy}(\tau)U_{xx}(\tau/2)U_0(\tau/2)]. \quad (19)$$

where  $\tau$  is the length of a single time interval. Figure 6 shows a circuit for performing the phase estimation algorithm. We will choose  $U$  to be an implementation of the BCS Hamiltonian for a single time interval in the Trotter approximation,  $U^2$  will require two time intervals,  $U^4$  will need four intervals etc. This will allow us to work out the number of operations required as a function of the number of ancilla qubits. We therefore find that the total number of operations required for running the local unitaries in the phase estimation algorithm with  $k$  ancillas is given by  $(2^k - 1)$  times the number of operations

needed to implement  $U$ , therefore we have

$$P_G(N) = (2^k - 1)(6N^2 + 64N - 40) \quad (20)$$

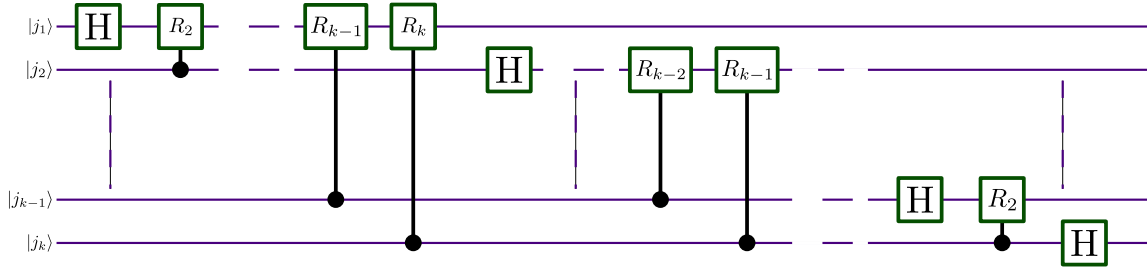
in the completely general case, and

$$P_L(N) = (2^k - 1)(12Np - 6p^2 - 6p + 70N - 40) \quad (21)$$

in the case of limited range interactions. The number of ancillas required is chosen dependent upon the desired accuracy.

## 6. Data extraction

As well as performing our controlled unitaries, the phase estimation algorithm requires us to implement the quantum Fourier transform (QFT), so we provide an efficient technique for performing the QFT on the qubus quantum computer. This provides us with significant savings over a naïve qubus implementation.



**Figure 9.** A circuit diagram for the quantum fourier transform on  $k$  qubits, where  $R_a$  are rotations on qubit  $a$ , and  $H$  is the Hadamard operation. The controlled unitaries which  $R_a$  are part of are described by the  $CR_a$  in equation (22).

The QFT involves performing controlled rotation operators on the qubits in a form shown in figure 9, where the gate  $CR_a$  corresponds to a controlled phase gate of the form

$$CR_a = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{2\pi i/2^a} \end{pmatrix}. \quad (22)$$

For each desired  $CR_a$  we perform an operation

$$CR'_a = \begin{pmatrix} e^{\pi i/2^{a+1}} & 0 & 0 & 0 \\ 0 & e^{-\pi i/2^{a+1}} & 0 & 0 \\ 0 & 0 & e^{-\pi i/2^{a+1}} & 0 \\ 0 & 0 & 0 & e^{\pi i/2^{a+1}} \end{pmatrix}. \quad (23)$$

We then correct these gates using local unitaries on each qubit, of the form

$$C_a = \begin{pmatrix} e^{-\pi i/2^{a+2}} & 0 \\ 0 & e^{3\pi i/2^{a+2}} \end{pmatrix}. \quad (24)$$

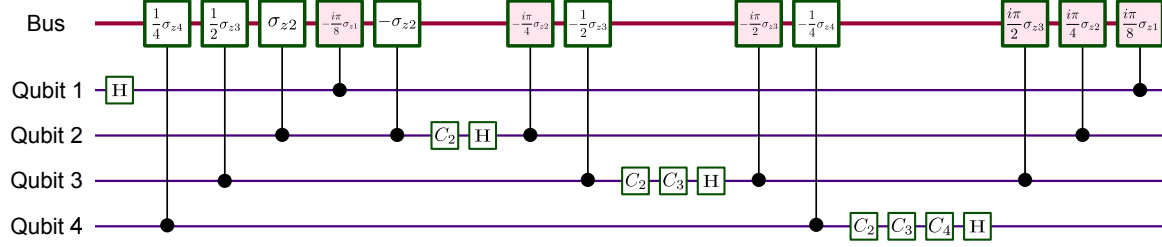
We chose to use this method of performing operations because of the limitations of the qubus quantum computer. The gates  $CR'_a$  are gates which are relatively easy to perform, and as we can apply general local unitaries the  $C_a$  gates don't present a problem.

Performing a QFT on  $k$  qubits involves a total of  $\frac{1}{2}(k^2 - k)$  controlled phase gates. In a naïve case where each phase gate requires 4 operations to perform, we would require  $2(k^2 - k)$  interactions with the bus. We can consider possible ways to reduce the total number of operations needed using a similar technique to the one described in section 3.2, since the size of the rotation we need to apply decreases exponentially with the separation of the qubits in the register. The principle differences between our technique here and our previous technique is the need to apply Hadamard gates between each set of controlled phase operations, and the local unitary corrections. As our corrections commute with the  $CR'_a$ , we don't need to perform these straight away. However, these corrections do not commute with the Hadamard, so the corrections for each individual qubit need to be performed before the Hadamard is performed on that qubit. Multiple corrections are performed simply by taking the product of the necessary corrections for every single controlled phase gate the qubit is part of, regardless of whether it acts as a target or control qubit.

To obtain the full QFT sequence we entangle qubit 1 with the first quadrature of the bus, then all the other  $k - 1$  qubits with the second quadrature. We then disentangle qubit 1 and qubit 2, before applying a correction and a Hadamard operation to qubit 2. Qubit 2 is then entangled to the first quadrature of the bus, thus interacting it with the other  $k - 2$  qubits. We repeat this process shifting successive qubits from the second quadrature of the bus to the first, applying a correction and a Hadamard operation in between the two operations. Once qubit  $k$  has been removed from the second quadrature of the bus, it doesn't need to be reconnected although the correction and the Hadamard operation need to be performed. We then remove qubits 2 through to  $k - 1$  from the first quadrature of the bus and our QFT is complete. In this sequence qubits 1 through to  $k - 1$  interact with the first quadrature of the bus twice (once to connect them, and once to remove them) and qubits 2 through to  $k$  interact with the second quadrature of the bus twice. This results in a total of  $4k - 4$  bus operations. A diagram with suitable phase factors for performing the QFT in the 4 qubit case is shown in figure 10.

We then need to consider the corrections we need to apply to our qubits. In the worst case scenario we would need to apply a correction for every single controlled rotation a qubit is part of, meaning we'd need  $k^2 - k$  local unitaries. Since we are considering diagonal matrices, and have the ability to perform arbitrary unitaries, it is possible to reduce this down to  $2k - 2$  by combining the corrections performed on each qubit into those performed before the Hadamard and those performed after. This is feasible because the matrices are diagonal, so we can work out what correction we need to apply efficiently on a classical computer. In the case of the phase estimation algorithm we are measuring in the Z-basis straight after performing our QFT. This means it is possible to reduce the number of corrections further by simply not performing those which occur after the Hadamard. This results in a total of  $k - 1$  corrections being





**Figure 10.** A diagram showing how to perform a 4 qubit Fourier Transform on the qubus quantum computer. The boxes show displacements performed on the continuous variable field controlled by the qubits or local operations. Shaded boxes represent an operation acting on the position quadrature of the bus, while unshaded boxes represent operations on the momentum quadrature.

applied and therefore a total number operations for the QFT given by

$$N_{FT} = 6k - 5 \quad (25)$$

including the  $k$  necessary Hadamard gates.

## 7. The total number of operations needed for a simulation

We now want to consider the total number of operations required for simulating the BCS Hamiltonian. The savings we showed in section 3 are mainly applicable to the initialisation procedure, however even for a small number of ancilla qubits the number of operations required by the phase estimation algorithm dominates to such a degree that it is only worth considering two cases, the general case and the case of limited range interactions. In the general case we find that the total number of operations needed to implement our entire algorithm for a single run is given by

$$T_G = P_G(N) + SI_G(N) + N_{FT} \quad (26)$$

from equations (14), (16), (20) and (25). This gives us

$$T_G = (2^k - 1)(6N^2 + 64N - 40) + 6k - 5 + \frac{0.1\pi}{\delta}(2N^2 + 3N + 4). \quad (27)$$

Using a similar argument for the limited range case we find,

$$T_L = (2^k - 1)(12Np - 6p^2 - 6p + 70N - 40) + 6k - 5 + \frac{0.1\pi}{\delta}(4pN + 5N - 2p^2 - 2p + 4). \quad (28)$$

To make a comparison with Wu et al. [17] we need to rewrite  $k$  in terms of precision. The level of precision available can be expressed as a function of  $2\pi$  such that the smallest phase difference we can detect is given by  $2\pi/2^k$ , to have an equivalent precision to Brown et al. [21] we want the smallest phase difference we can detect to be given by  $\delta$ , therefore  $2^k = 2\pi/\delta$ . The total number of operations is seen to be

$$T_{Gk} \approx \frac{0.1\pi}{\delta}(122N^2 + 1283N - 796) \quad (29)$$

in the general case, and

$$T_{\text{Lk}} \approx \frac{0.1\pi}{\delta}(244Np - 122p^2 - 122p + 1405N - 796) \quad (30)$$

in the limited range case. We can now compare our results to those found by Wu et al. [17] who claim to require significantly more than  $3N^4$  operations, however this excludes the initialisation procedure. Since they use the first order Trotter approximation throughout we will derive an upper bound by using the same number of operations for the initialisation procedure as the longest run of the computer. We will incorporate the precision such that

$$T_{\text{NMR}} = \frac{6}{\delta}N^4 \quad (31)$$

in the nearest neighbour case to leading order. For our qubus system in the nearest-neighbour case we have

$$T_{\text{qubus}} = \frac{\pi}{\delta}(164.7N - 104). \quad (32)$$

We can therefore see that provided  $N \geq 5$  our qubus system requires less operations than an equivalent NMR simulation. For  $N < 5$  it would be possible to get savings by using a similar data extraction procedure to NMR, running the simulation for several time intervals and working out the probability of a single qubit being in  $|1\rangle$ . While this would require more runs and have a poor scaling it is suitable for small systems. In the case of  $N = 10$  our qubus system requires  $785,430 \approx 8 \times 10^5$  operations, while the NMR system requires  $6 \times 10^6$  operations, therefore we can already see a significant difference. Using a similar number ( $\approx 6 \times 10^6$ ) of operations on our qubus system it would be possible to generate operations for a BCS Hamiltonian of 72 qubits in the nearest-neighbour case, and 26 qubits in the general case.

## 8. Conclusions

In this paper we have discussed how to simulate the BCS Hamiltonian on a qubus quantum computer. Our results are also applicable to general ancilla-based schemes. We show that using a naïve method of performing the time evolution, we require  $O(N^2)$  operations, which is an  $O(N^3)$  saving over a simulation on an NMR computer [17, 21]. It is possible to get further reductions in the number of gates needed in the general case, where we get almost a factor of 2 savings over our naïve method, and in certain special cases where  $O(N)$  saving is possible. In the general case we need  $3N^2 + 2N + 6$  operations, including local unitaries, for each time step of the evolution. For the special cases, which include the case where interactions are decaying exponentially with distance, only  $17N - 12$  operations are required for each stage in the time evolution. We show that the general case requires only 6 operations more than our lower bound, and that our special cases achieve their overall lower bounds. We also look at the nearest-neighbour case of the BCS Hamiltonian which only requires  $2N$  operations for  $U_{zz}$  [25] so  $11N$  operations to perform in the general case. Building upon this we look at interactions

of length  $p$ , where  $p = 1$  is the nearest-neighbour case,  $p = 2$  nearest-neighbour and next to nearest-neighbour etc. For this case, we characterise the number of operations needed to perform each stage of the time evolution as  $5N + 6pN - 3p^2 - 3p + 6$ .

We then apply these efficiency savings to making our operations controlled so we can implement the phase estimation algorithm. In this case we find that we can't get the  $O(N)$  savings over our naïve method. However, we can still obtain significant savings, and require  $O(N^3)$  less operations than an NMR computer. As we are using the phase estimation algorithm we also demonstrate how to obtain efficiency savings when performing a QFT on the qubus quantum computer, reducing the number of operations required by  $O(N)$  to  $6N - 5$  per QFT. This significant improvement will allow efficient extraction of the energy gap from our system using the phase estimation algorithm. The efficiency savings for the quantum Fourier transform are applicable to other quantum algorithms, such as Shor's algorithm.

Wu et al. [17] give a minimum bound on the number of operations required to simulate a BCS Hamiltonian described by 10 qubits on an NMR quantum computer. If we consider being limited to a similar number of operations, we show that on the qubus quantum computer we can simulate a BCS Hamiltonian described by 72 qubits. Thus our qubus quantum computer has significantly better efficiency than the NMR computer, and hence we will be able to increase the size of the system we can simulate on an early quantum computer.

We have shown that the qubus quantum computer has significant advantages in the number of elementary operations required compared to an NMR computer, this gives  $O(N^3)$  savings when simulating the BCS Hamiltonian. As we have demonstrated an  $O(N)$  improvement over the naïve case of performing the quantum Fourier transform we have every reason to believe that these savings will apply in a wide range of cases. In other work we have shown similar improvements for generating cluster states [27], illustrating the generality of the techniques we have described here in the context of simulating the BCS Hamiltonian.

## Acknowledgments

KLB is supported by a UK EPSRC CASE studentship from Hewlett Packard. VMK is funded by a UK Royal Society University Research Fellowship. WJM was supported in part by MEXT and FIRST in Japan. We would like to thank Tim Spiller for useful discussions on the BCS Hamiltonian and the qubus architecture.

## References

- [1] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, October 1997.
- [2] Richard P. Feynman. Simulating physics with computers. *Internat. J. Theoret. Phys.*, 21(6/7):467–488, June 1982.
- [3] Seth Lloyd. Universal quantum simulators. *Science*, 273(5278):1073–1078, August 1996.

- [4] Alan Aspuru-Guzik, Anthony D. Dutoi, Peter J. Love, and Martin Head-Gordon. Simulated quantum computation of molecular energies. *Science*, 309(5741):1704–1707, September 2005.
- [5] Smirnov, S. Savel'ev, L. G. Mourokh, and Franco Nori. Modelling chemical reactions using semiconductor quantum dots. *EPL*, 80(6):67008, December 2007.
- [6] Daniel S. Abrams and Seth Lloyd. Quantum algorithm providing exponential speed increase for finding eigenvalues and eigenvectors. *Phys. Rev. Lett.*, 83(24):5162, December 1999.
- [7] Katherine L. Brown, William J. Munro, and Vivien M. Kendon. Using quantum computers for quantum simulation. *Entropy*, 12(11):2268–2307, November 2010.
- [8] K. De Raedt, K. Michielsen, H. De Raedt, B. Trieru, G. Arnold, M. Richter, Th. Lippert, H. Watanabe, and N. Ito. Massive parallel quantum computer simulator. *Comp. Phys. Comm.*, 176(2):121–136, January 2007.
- [9] F. Verstraete, D. Porras, and J. I. Cirac. Density matrix renormalization group and periodic boundary conditions: A quantum information perspective. *Phys. Rev. Lett.*, 93:227205, 2004.
- [10] J. Bardeen, L. N. Cooper, and J. R. Schrieffer. Theory of superconductivity. *Phys. Rev.*, 108(5):1175, December 1957.
- [11] R. W. Richardson and N. Sherman. Exact eigenstates of the pairing-force Hamiltonian. *Nuclear Physics*, 52:221–238, March/April 1964.
- [12] P. W. Anderson. Random-phase approximation in the theory of superconductivity. *Phys. Rev.*, 112(6):1900–1916, December 1958.
- [13] A. Mastellone, G. Falci, and Rosario Fazio. Small superconducting grain in the canonical ensemble. *Phys. Rev. Lett.*, 80(20):4542–4545, May 1998.
- [14] J. Dukelsky and G. Sierra. Density matrix renormalization group study of ultrasmall superconducting grains. *Phys. Rev. Lett.*, 83(1):172–175, July 1999.
- [15] Alexander Volya, B. Alex Brown, and Vladmimir Zelevinsky. Exact solution of the nuclear pairing problem. *Phys. Lett. B*, 509(1/2):37–42, June 2001.
- [16] S. Y. Ho, D. J. Rowe, and S. De Baerdemacker. Eigenstate Estimation for the Bardeen-Cooper-Schrieffer (BCS) Hamiltonian. arXiv:1011.4304, November 2010.
- [17] L.-A. Wu, M. S. Byrd, and D. A. Lidar. Polynomial-time simulation of pairing models on a quantum computer. *Phys. Rev. Lett.*, 89(5):057904, July 2002.
- [18] Richard Cleve, Artur Ekert, Chiara Macchiavello, and Michele Mosca. Quantum algorithms revisited. *Proc. Roy. Soc. London A*, 454(1969):339, January 1998.
- [19] P. van Loock, W. J. Munro, Kae Nemoto, T. P. Spiller, T. D. Ladd, Samuel L. Braunstein, and G. J. Milburn. Hybrid quantum computation in quantum optics. *Phys. Rev. A*, 78(2):022303, August 2008.
- [20] T. P. Spiller, Kae Nemoto, Samuel L. Braunstein, W. J. Munro, P. van Loock, and G. J. Milburn. Quantum computation by communication. *New Journal of Physics*, 8(2):30, February 2006.
- [21] Kenneth R. Brown, Robert J. Clark, and Issac L. Chuang. Limitations of quantum simulation examined by a pairing Hamiltonian using nuclear magnetic resonance. *Phys. Rev. Lett.*, 97(5):050504, August 2006.
- [22] D. A. Rodrigues, C. E. A. Jarvis, B. L. Gyroff, T. P. Spiller, and J. F. Annett. Entanglement of superconducting charge qubits by homodyne measurement. *Journal of Physics: Condensed Matter*, 20(7):075211, January 2008.
- [23] H. F. Trotter. On the product of semi-groups of operators. *Proc. Am. Math. Phys.*, 10:545, 1959.
- [24] Masuo Suzuki. Improved Trotter-like formula. *Phys. Lett. A*, 180(3):232 – 234, 1993.
- [25] Sebastien G. R. Louis, Kae Nemoto, W. J. Munro, and T. P. Spiller. The efficiencies of generating cluster states with weak non-linearities. *New Journal of Physics*, 9(6):193, June 2007.
- [26] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. Camb, 2000.
- [27] Clare Horsman, Katherine L. Brown, William J. Munro, and Vivien M. Kendon. Reduce, reuse, recycle, for robust cluster state generation. arXiv:1005.1621v2, Jul 2010.

